REMARKS

Applicant is in receipt of the Office Action mailed June 25, 2004. Claims 1-17 are pending in the Application. Claim 1 was amended to more clearly characterize the claimed invention. Further consideration of the present case is earnestly requested in light of the following remarks.

§ 103 Rejections

Claims 1-17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lentz (U.S. Parent No. 5,515,494) in further view of Duluk (U.S. Patent No. 6,552,723). Applicant respectfully traverses these rejections based on the following reasoning.

Claim 1

Claim 1 of the Application recites:

> "A method for comparing a pixel against one or more windows using a pipeline, the method comprising:
>
> passing said pixel through the pipeline, wherein said pipeline comprises two or more pipeline segments;
>
> computing a window result in each one of said two or more pipeline segments, wherein each one of said two or more pipeline segments corresponds to one of said one or more windows, wherein said window result comprises an indication of inclusion of said pixel within the corresponding one of said one or more windows;
>
> outputting a window word from each one of said two or more pipeline segments, wherein said outputting said window word comprises, for each of said two or more pipeline segments except for a last pipeline segment, passing said window word to a next pipeline segment, wherein said window word comprises said window result and a previous window result; and
>
> examining said window word available in the last pipeline segment to determine if the pixel is included in the one or more windows."

In contrast, Lentz teaches a set of graphics control planes used to control video features in the graphics system, including a write enable plane. Lentz describes use of a window data structure for defining imported window parameters, which is used to determine whether a window in which an object to be drawn is obscured by another window. This information is used to determine if pixels for that window are discarded.

Although superficial similarities exist between claim 1 of the Application and the system of Lentz, the differences between the two systems will be set forth herein. As the Office Action notes, Lentz does not teach or suggest using a pipeline for multiple pixel operations in succession. Instead, as discussed *infra*, Lentz teaches examining and indicating visibility of windows relative to other windows. In the system of Lentz, a write enable plane is used to indicate whether a pixel is within a visible portion of an active window. The write enable plane of Lentz is used as a part of a larger data structure used to control video features in a graphic system of Lentz. In contrast, claim 1 of the Application discloses computing a window result in each one of the pipeline segments, wherein each one of the pipeline segments corresponds to one the windows, wherein the window result includes an indication of inclusion of said pixel within the corresponding one the one or more windows.

The Office Action cites Lentz col. 11, lines 35-42 as comparing a pixel against one or more windows, the method comprising: passing said pixel data, computing window result in each computation, wherein each computation corresponds to one of said one or more windows. The Applicant respectfully disagrees. As discussed *infra*, Lentz discloses a system for determining and indicating visibility of windows relative to other windows.

Lentz in column 11, lines 35-42 discloses a part of a method to use a hardware write enable test:

> "In step 714, the hardware write enable test for each window is set to correspond to the convention chosen for each window. The test is set such that for unobscured regions of the window the test is true. For example, the presence of ones in write enable plane 328 within the boundaries of window 402 indicate that information for window 402 is to be written to frame buffer 204. Thus the test is true for window 402 when a one appears in write enable plane 328. "

In other words, Lentz in step 714 discloses a way to determine if a window is obscured by other windows. Specifically, in column 12, lines 1-23, Lentz gives an exemplary use of the method of Figure 7:

> "FIG. 9 provides an example illustration of the bits in write enable plane 328 for the windows illustrated in FIG. 4 according to the first embodiment. In window 402 ones indicate portions of the window that are unobscured while zeros indicate active portions.

> In a further embodiment, the convention used by write enable plane 328 for every window is the same. For example, a one indicates unobscured regions for every window, while a zero indicates obscured regions for every window. In this further embodiment, if the display controller writes to a first window in a first operation, and wants to write to a second window in a second operation, where the windows overlap, write control plane 328 must be updated.
>
> For example, suppose graphics server 206 writes to window 404. To enable this operation, all the bits in write enable plane 328 within window clip boundary 500 for window 404 are set to the value of one. If display controller 202 next wants to write to window 402, the bits in write enable plane 328 corresponding to region 442 (where window 404 is on top of 402) must be changed from ones (used to write in previous operation) to zeros. Thus, according to the second embodiment, step 712 must be performed regardless of whether the window definition changed.

In other words, the method of Lentz operates to examine and indicate the visibility of windows relative to other windows, as shown in Figures 4, 6, and 9, beside others. Figure 9 of Lentz illustrates a write control plane, which shows the location of bits that indicate the visibility/obscurity of windows and portions of windows of Figure 4. This is significantly different from processing a pixel by computing a window result in each one of computations to indicate inclusion of the pixel within a corresponding one or more windows and examining a window word available in a last computation to determine the pixel's inclusion in any of the windows as disclosed in claim 1 of the Application.

Furthermore, the Office Action points to col. 12, lines 23-35 of Lentz as disclosing "wherein said window result comprises an indication of inclusion of said pixel within the corresponding one of said one or more windows." The Applicant respectfully disagrees. Lentz in col. 21, lines 23-35 discloses:

> "FIG. 10 is a flowchart illustrating the method according to the second embodiment of the invention. Graphics server 206 determines if a window is completely obscured in a step 1002. This determination is based on data structure 600. If the window is completely obscured, the operation ceases. Graphics server 206 determines whether the window is partially obscured in a step 1004. If the window is not partially obscured (i.e., it is totally unobscured), in a step 1006, display controller 202 uses window clip boundaries 500 to determine whether a pixel is to be written to frame buffer 204. Display controller writes the appropriate pixels to frame buffer 204 in a step 1012. These pixels fall within a visible portion of the active window."

In other words, in the passage above Lentz describes a method for determining if a window is obscured. Lentz further discusses that if the window is not obscured, then the display controller uses window clip boundaries to determine whether a pixel is to be written to the frame buffer. Put yet another way, the system of Lentz is concerned about determining which windows or parts of windows are obscured and which windows or parts of windows are visible. This is significantly different from a window result that includes an indication of inclusion of a pixel within the corresponding one of the one or more windows as disclosed in claim 1 of the Application.

Furthermore, the Office Action states that Lentz in column 6, lines 23-25 teaches outputting a window word from each computation, wherein said outputting said window word comprises, for each of computations except for a last computation, passing said window word to next computation. Applicant respectfully disagrees. Lentz in column 6, lines 23-25 discloses:

> "Thus, using WIDs, only pixels in the foreground windows for each pixel location are written to the frame buffer."

Actually, in the context of the Lentz patent, the above passage refers to:

> "To write information to a display memory, the WID for each pixel of the frame buffer is stored in a WID register 106 and compared to the WID in window mapping buffer 104 for that pixel location. If the WID in window mapping buffer 104 is the same as the WID for the pixel to be displayed, a comparator circuit 108 causes write enable logic to allow the pixel information to be written to a frame buffer 110 at the correct pixel address. If, on the other hand, comparator circuitry 108 determines that the WID for the pixel is not the same as the WID stored in window mapping buffer 104, then the pixel information is not written to frame buffer 110. Consequently, only those pixels of a particular location belonging to a window to be displayed at that location are written to frame buffer 110 and subsequently displayed." (Lentz col. 6, lines 10-24)

The WID described above are "additional planes included to store a code identifying a particular window to which each pixel belongs." (Lentz col. 5, lines 36-40).

Furthermore, WID are described in col. 2, lines 42-58:

> "Conventional graphics systems use window ID's (WIDs) to control drawing into windows. Before defining WIDs, it will be useful to define the term "plane" as it is used in frame buffer applications. A plane is a subset of the pixmap comprising

the same bit in all pixels. A plane is a "horizontal" cross-section of the pixmap. Thus, the set consisting of the first bit in each pixel, for example, makes up a plane.

In a system employing WIDs, additional planes are included in the frame buffer memory. The additional planes hold a code, called a WID. The WID code identifies the window to which a pixel belongs. When the pixel is sent to the frame buffer, its WID code is compared to the WID identifying the window to be displayed at that location on the screen. If the comparison agrees, a write enable signal is generated and the pixel is saved in the frame buffer. If the WIDs do not compare, no write enable signal is generated and that pixel is not written to the frame buffer."

In other words, WID's are Window ID's for a pixel that identify to which window identify which the window to which a pixel belongs. In context of the passage in column 6, lines 23-25, by using WID's a system operates to examine which pixels are in foreground windows. Furthermore, the Office Action equates data for included pixels that are passed to a frame buffer to outputting a window word from each computation, including, for each of computations except for a last computation, passing the window word to next computation. Applicant respectfully disagrees. In the system of Lentz, as disclosed above, the pixels that are passed to the frame buffer have an associated WID code that is compared to a WID code for a window to be displayed on screen. Applicant notes that the comparison of WID codes, i.e., where the WID code for the pixel is examined to match the WID code of a window to be displayed, is significantly different from outputting a window word from each computation, including, for each of computations except for a last computation, passing the window word to next computation. Furthermore, Applicant notes that the use of WIDs and of graphic control planes, e.g., a write enable plane, is mutually exclusive, as noted in column 7, lines 8-11 of Lentz:

"The present invention eliminates the need for WID planes by providing an alternative set of planes, called graphics control planes."

Therefore it is not proper to combine the WID planes and the graphic control planes of Lentz.

Additionally, the Office Action equates examining the window word available in the last computation to the determination described in col. 10, lines 5-65 of Lentz:

> "Turning now to FIG. 7, in a step 702, graphics screen 206 first determines whether the window in which an object is to be drawn is completely obscured. If it is obscured totally, the operation ends and all drawing operations may be discarded since the window is not visible. Drawing can be discarded in many ways, but the simplest is to not pass the data to frame buffer 204. If the window is not totally obscured the operation continues at a step 704. This determination is made using data structure 600."

As discussed above, Figure 7 of Lentz operates to determine obstructions of windows relative to each other. Then, using window clip boundaries and write enable plane the system is operable to determine if the pixel is to be written to a frame buffer. The write enable plane of Lentz is illustrated in Figure 9 as described *supra*. In addition, Applicant respectfully notes that the examining the window word available in the last computation of claim 1 of the Application is done in response to data for included pixels that are passed to a frame buffer to outputting a window word from each computation, including, for each of computations except for a last computation, passing the window word to next computation. In contrast, the description of Figure 7 of Lentz describes determination if a window in which an object is to be drawn is completely obscured.

Furthermore, Applicant respectfully notes that it is not proper to combine two or more references when there is no teaching or motivation to do so. As held by the *U.S. Court of Appeals for the Federal Circuit in Ecolochem Inc. v. Southern California Edison Co.*, an obviousness claim that lacks evidence of a suggestion or motivation for one of skill in the art to combine prior art references to produce the claimed invention is defective as hindsight analysis.

In addition, the showing of a suggestion, teaching, or motivation to combine prior teachings "must be clear and particular . . .. Broad conclusory statements regarding the teaching of multiple references, standing alone, are not 'evidence'." *In re Dembiczak*, 175 F.3d 994, 50 USPQ2d 1614 (Fed. Cir. 1999). The art must fairly teach or suggest to one to make the specific combination as claimed. That one achieves an improved result

by making such a combination is no more than hindsight without an initial suggestion to make the combination.

Specifically, Duluk describes a mid-pipeline sorting unit that sorts image data mid-pipeline in a tiled 3-D graphics pipeline architecture. This is significantly different from a pipeline operable to pass a pixel through a pipeline and compute a window result in each one of the pipeline segments which indicates inclusion of the pixel within the corresponding one or more windows and examine a window word available in the last pipeline segment to determine the pixel's inclusion in any of the windows as disclosed in claim 1 of the Application. Applicant respectfully notes that neither Lentz nor Duluk provide motivation to combine. Furthermore, as discussed *supra*, even if the graphical pipeline of Duluk was combined with the teachings of Lentz, which is not taught or suggested by either Lentz or Duluk, it would not produce the method of claim 1, as the teachings of Lentz are significantly different from the method of claim 1.

In conclusion, Lentz and Duluk, taken singly or in combination, do not teach or suggest passing a pixel through a pipeline and computing a window result in each one of the pipeline segments which indicates inclusion of the pixel within the corresponding one or more windows and examining a window word available in the last pipeline segment to determine the pixel's inclusion in any of the windows. Similar arguments apply with equal force to independent claims 8 and 15. Applicant thus respectfully submits that each of the independent claims 8 and 15, and claims dependent thereon, are patentably distinct over the cited art, and are thus allowable. Thus, Applicant respectfully requests that the section 103 rejection of claims 1-17 be removed.

Applicant also asserts that numerous ones of the dependent claims recite further distinctions over the cited art. However, since the independent claims have been shown to be patentably distinct, a further discussion of the dependent claims is not necessary at this time.

CONCLUSION

Applicant submits the application is in condition for allowance, and an early notice to that effect is requested.

If any extensions of time (under 37 C.F.R. § 1.136) are necessary to prevent the above referenced application(s) from becoming abandoned, Applicant(s) hereby petition for such extensions. If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert & Goetzel PC Deposit Account No. 50-1505/5181-84600/JCH.

Also enclosed herewith are the following items:

☒ Return Receipt Postcard

Respectfully submitted,

Jeffrey C. Hood
Reg. No. 35,198
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert & Goetzel PC
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8800
Date: June 25, 2004